

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 02-217925

(43)Date of publication of application : 30.08.1990

(51)Int.Cl.

G06F 9/42

G06F 9/46

(21)Application number : 01-037805

(71)Applicant : TOSHIBA CORP

TOSHIBA MICRO ELECTRON KK

(22)Date of filing : 17.02.1989

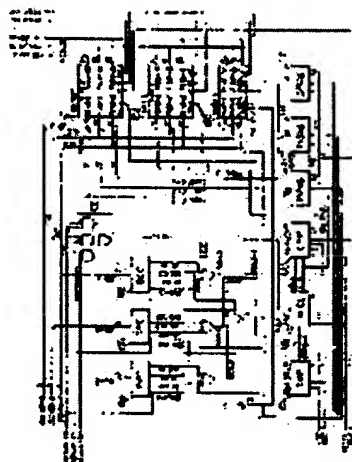
(72)Inventor : SAKAGAMI KENJI

## (54) MICROPROCESSOR

## (57)Abstract:

PURPOSE: To improve cost performance by flexibly determining the number of registers to be allocated to windows in accordance with the number of registers to be used for a procedure in a program

CONSTITUTION: A microprocessor is constituted of R-FILE1 modules 101, 102, an R-FILE2 module 103, decoders(DECs) 104 to 106, a current window pointer (CWP) 107, a register(W-CC) 108, a current block pointer (CBP) 109, and substructors(INSAGs) 110 to 112. A working register to be used for each procedure is determined in accordance with the individual procedure in the program and the number of registers constituting the working register is determined. Consequently, the redundancy of the system is removed and the cost performance is improved.



## LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

BEST AVAILABLE COPY

⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A) 平2-217925

⑬ Int. Cl.<sup>3</sup>

G 06 F 9/42  
9/46

識別記号

3 3 0 R  
3 1 3 C

庁内整理番号

7361-5B  
8945-5B

⑭ 公開 平成2年(1990)8月30日

審査請求 未請求 請求項の数 2 (全12頁)

⑮ 発明の名称 マイクロプロセッサ

⑯ 特 願 平1-37805

⑰ 出 願 平1(1989)2月17日

⑱ 発 明 者 坂 上 健 二 神奈川県川崎市川崎区駅前本町25番地1 東芝マイコンエンジニアリング株式会社内

⑲ 出 願 人 株 式 会 社 東 芝 神奈川県川崎市幸区堀川町72番地

⑲ 出 願 人 東芝マイクロエレクトロニクス株式会社 神奈川県川崎市川崎区駅前本町25番地1

⑳ 代 理 人 弁 理 士 佐 藤 一 雄 外3名

明 細 書

1. 発明の名称

マイクロプロセッサ

2. 特許請求の範囲

1. 複数個のレジスタウィンドウを有し、前記各レジスタウィンドウは複数個のレジスタを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウィンドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウィンドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウィンドウへのレジスタ割り当て数を、前記ワーキングレジスタに対するアドレス情報に基づいて自動的に記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウィンドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えることを特徴とするマイクロプロセッサ。

2. 複数個のレジスタウィンドウを有し、前記各レジスタウィンドウは複数個のレジスタを有し、さらにCPUを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウィンドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウィンドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウィンドウへのレジスタ割り当て数を、前記CPUが実行できる命令の1つに基づいて記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウィンドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えることを特徴とするマイクロプロセッサ。

### 3. 発明の詳細な説明

#### (発明の目的)

##### (産業上の利用分野)

本発明は、マイクロプロセッサに関し、特に、RISC (Reduced Instruction Computer Set)タイプのマイクロプロセッサに関する。

##### (従来の技術)

SPARCアーキテクチャによるマイクロプロセッサは、例えば、刊行物「10 MIPSから100 MIPSまでを単一アーキテクチャでカバーするSPARC」(Nikkei Byte/MARCH 1988)に示されている。ここには、レジスタウィンドウを用いたマイクロプロセッサに関して詳述されている。このようなシステムで使用する、プロシー

ジャコールを多用するプログラムにおいては、多くのウィンドウを使用することになる。

従って、マイクロプロセッサチップ上のウィンドウ数が少ないとウィンドウのオーバーフロー/アンダーフロー処理によるオーバーヘッドが大きな問題となってくる。SPARCアーキテクチャでは、1つのウィンドウを追加するためには、32ビット長のレジスタを16本必要とする。しかしながら、チップ上のウィンドウ数はハードウェア量によって制限を受けるため、現実には多くすることはできない。ちなみに、SPARCでは、ウィンドウの数は最小6個、最大32個と規定しているが、1.5  $\mu$ m CMOSゲートアレイによるチップの数は、上記刊行物では7個となっている。

##### (発明が解決しようとする課題)

従来のマイクロプロセッサは以上のように構成されているので、1個のウィンドウに割り当てるレジスタ数は固定である(以下これを固定方式と呼ぶ)。しかしながら、個々のプロシージャコー

ルで実際に使用するレジスタ数は割り当て数より少ない場合も多くあり、割り当て数が固定であるということは非常に冗長である。このため、例えば、現在のウィンドウで未使用のレジスタを別のウィンドウに割り当ててウィンドウ数を増やすといった、レジスタの有効活用ができない。

このような問題を解決する目的で、1つのウィンドウに割り当てるレジスタ数を可変にする方式が従来技術として知られている(以下、この方式を可変方式と呼ぶ)。

本発明は、可変方式のレジスタウィンドウを含むマイクロプロセッサの構成技術に関し、その目的の1つは、プロセッサの実行プログラムの内容に応じて、自動的に1つのウィンドウに割り当てるレジスタ数を決定する可変方式のレジスタウィンドウの構成技術を提供し、固定方式のレジスタウィンドウによるマイクロプロセッサのみに開発されたプログラムをそのまま実行できる可変方式のレジスタウィンドウによるプロセッサの構成技術を提供することにより、上記プログラムの内容

の変更なしに従来のプログラムの処理速度を、ウィンドウのオーバーフロー/アンダーフローのオーバーヘッドを低減することにより、改善出来るようにすることにある。

目的のもうひとつは、可変方式のレジスタウィンドウによるマイクロプロセッサで、その命令セットの中に、ウィンドウへのレジスタ割り当て数を制御する命令を設けたマイクロプロセッサ技術を提供し、プログラム内のプロシージャの使用するレジスタ数の規模に応じて柔軟にウィンドウへのレジスタ割り当て数を決定できるようにし、使い勝手が良く、かつウィンドウの使用効率を高め、ウィンドウのオーバーフロー/アンダーフローによるオーバーヘッドを低減でき、コストパフォーマンスの優れたマイクロプロセッサを提供することにある。

##### (発明の構成)

##### (課題を解決するための手段)

本発明の第1のマイクロプロセッサは、複数個のレジスタウィンドウを有し、前記各レジスタウ

インドウは複数個のレジスタを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウインドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウインドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウインドウへのレジスタ割り当て数を、前記ワーキングレジスタに対するアドレス情報に基づいて自動的に記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウインドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えるものとして構成される。

本発明の第2のマイクロプロセッサは、複数個のレジスタウインドウを有し、前記各レジスタウインドウは複数個のレジスタを有し、さらにCPU

Uを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウインドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウインドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウインドウへのレジスタ割り当て数を、前記CPUが実行できる命令の1つに基づいて記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウインドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えるものとして構成される。

(作 用)

各レジスタウインドウ(モジュール)へのレジスタ割り当て数が記憶手段に記憶されている。第2決定手段は、その割り当て数に応じて、各モジ

ュールがどのレジスタによって、即ち何個のレジスタによって構成されるかを決定する。従って、記憶手段中の割り当て数によって、各モジュールが何個のレジスタによって構成されるかが変化することとなる。即ち、プログラム上の個々のプロシージャに応じて、そのプロシージャ上で使用されるワーキングレジスタが第1決定手段で決定されると共にワーキングレジスタを構成するレジスタの数が第2決定手段で決定される。

さらに、第1の発明においては、記憶手段での、各レジスタウインドウへのレジスタ割り当て数の記憶は、ワーキングレジスタに対するアドレス情報に基づいて自動的に行われる。また、第2の発明においては、前記割り当て数の記憶は、CPUの実行できる命令の1つに基づいて行われる。

(実施例)

以下、図面を参照しながら本発明の実施例を説明する。

第1図は本発明の一実施例に係るマイクロプロセッサのブロック図で、特にウインドウの

locals、outsに割り当てるレジスタ数を8本または4本とすることを可能とした例を示すものである。

第1図において、レジスタアドレスrs1-A0~4はソースレジスタ1を、レジスタアドレスrs2-A0~4はソースレジスタ2を、レジスタアドレスrd-A0~4はデステネーションレジスタをそれぞれ指定するものである。前記各アドレスのうちのA0~2はそれぞれR-FILE1モジュール101、R-FILE1モジュール102、R-FILE2モジュール103に与えられる。より詳しくは、アドレスrs1-A0~2は各モジュールのポートP0-A0~6に、アドレスrs2-A0~2はポートP1-A0~6に、アドレスrd-A0~4はポートP2-A0~6にそれぞれ与えられる。また、各アドレスA2は論理ブロック100に与えられる。そして、各アドレスのA3~4はそれぞれデコード(DEC)104、105、106に与えられる。一方、rs1-D0~31はソースレジスタ1

の出力データであり、 $r_{s2-D0\sim31}$ はソースレジスタ2の出力データである。これらの $r_{s1-D0\sim31}$ および $r_{s2-D0\sim31}$ のいずれもR-FILE1モジュール101、R-FILE1モジュール102、R-FILE2モジュール103のそれぞれのポートP0-D0~31、P1-D0~31から出力される。また、 $r_{d-D0\sim31}$ はデステネーションレジスタへの入力データであり、R-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103の各ポートP2-D0~31に与えられる。

なお、A0~4で示されるアドレスはglobals 8本と1個のウィンドウ24本 (ins 8本、locals 8本、outs 8本) の計32本のワーキングレジスタの指定を行なうものである。D0~31で示されるデータはアドレスA0~4に対応するワーキングレジスタの入出力データである。

なお、プログラム上の基本演算は2つのソースレジスタと1つのデステネーションレジスタとの

間で見かけ上1サイクルで行なうようにされている。ワーキングレジスタを構成するR-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103は、データ出力ポートが2個、データ入力ポートが1個の3ポート構成となっている。各アドレスおよびデータは、R-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103の各ポートに対して独立に接続されている。

R-FILE1モジュール101はins/outsのワーキングレジスタを構成するレジスタファイルのモジュールで、32ビット×56本の構成を有する。R-FILE1モジュール102はlocalsのワーキングレジスタを構成するレジスタファイルのモジュールで、32ビット×56本の構成を有する。また、R-FILE2モジュール103はglobalsのワーキングレジスタを構成するレジスタファイルのモジュールで、32ビット×8本の構成を有する。

ちなみに、各R-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103において、ΦはクロックCLKの入力端子、REはリードイネーブルREGR-ENの入力端子、WEはライトイネーブルREGW-ENの入力端子、S0~2は各ポートの選択を行なう信号の入力端子である。

R-FILE1モジュール101の内部は、第2図に示すように、0~13の番地付けのなされた14個のブロックに分割されている。さらに、1つのブロックは4本のレジスタにより構成される。そして、ポートP0のアドレスA3~6 (上位4ビット) がブロックアドレスになり、ポートP0のアドレスA0~2 (下位3ビット) がポートP0のアドレスA3~6が指定するブロックをベースとした8本のレジスタに対するアドレスになる。ポートP1のアドレスA0~6、ポートP2のアドレスA0~6も上記と同様である。ウィンドウ13が8本モードであるときのレジスタファイルの割り当てブロックは、localsはR-

FILE1モジュール102のブロック13、12、outsはR-FILE1モジュール101のブロック13、12、insはR-FILE1モジュール101のブロック0 (初期状態) である。また、ウィンドウ13が4本モードであるときのレジスタファイルの割り当てブロックはlocalsはR-FILE1モジュール102のブロック13、outsはR-FILE1モジュール101のブロック13、insはR-FILE1モジュール101のブロック0 (初期状態) となる。この他、S0~2の入力信号によってポートP0~2がイネーブルかそうでないかを決定する。ちなみに、ポートP0~2がイネーブルとなるのはS0~2の入力信号が“1”の時である。

また、カレントウィンドウポインタ (CWP) 107は現在のプロシージャが使用しているウィンドウの番号を出力するものであり、且つインクリメント信号INCに基づいてインクリメント可能な5ビットのバイナリカウンタである。ちなみに、SPARCではカレントウィンドウポインタ

(CWP) はプロセッサステートレジスタ (PSR) の下位 5 ビットを形成しているので、本実施例ではこれに準じて 5 ビットとしている。このカレントウインドウポインタ (CWP) 107 は、リセット信号 RESET が入力されると、カウント値がプロシージャが最初に使用するウインドウの番号 CWP = 13 にセットされる。ちなみに、SPARC ではウインドウ番号の最も大きなものから使用していくので、本実施例はこれに準じている。なお、CWP = 0 の状態でディクリメント信号 DEC によりディクリメントすると CWP = 13 に戻る。そして、カレントウインドウポインタ (CWP) 107 の出力信号 CWP0 ~ 4 は次に説明するレジスタ (W-CC) 108 のビットを指定する。

レジスタ (W-CC) 108 は、各ウインドウが、8 本モードかあるいは 4 本モードかの情報を記憶する 14 ビットのレジスタである。14 ビットはそれぞれ 14 個のウインドウに対応している。各ビットは "0" が 4 本モードに、"1" が 8 本

モードにそれぞれ対応している。これは、RESET が入力されると全ビットが "0" にリセットされ、4 本モードに設定される。8 本モードへの切り替えは、次のようにして自動的に実施される。即ち、outs と locals のワーキングレジスタのそれぞれの 8 個のアドレスの順に前半と後半の 4 個に分ける。プロシージャで outs、locals と後半の 4 個しかアクセスせずに次のウインドウに移った場合は、そのウインドウは 4 本モードのままになっている。もし、outs、locals のいずれかで、前半のいずれかのレジスタをアクセスすると、レジスタ (W-CC) 108 の対応するビットが "1" にセットされ、このウインドウは 8 本モードとなる、locals の前半は r16 ~ r19、後半は r20 ~ r23、outs の前半は r8 ~ r11、後半は r12 ~ r15 である。

一方、許可信号 rslccw-en は、ソースレジスタ 1 のレジスタアドレスの rsl-a2 を、レジスタ (W-CC) 108 のセットに使用することを許可するものである。また、許可信号 rs

2ccw-en は、ソースレジスタ 2 のレジスタアドレスの rs2-a2 を、レジスタ (W-CC) 108 のセットに使用することを許可するものである。そして、許可信号 rdccw-en は、デスチネーションレジスタのレジスタアドレスの rd-a2 を、レジスタ (W-CC) 108 のセットに使用することを許可するものである。これらの許可信号 rslccw-en、rs2ccw-en、rdccw-en は各アドレス A2 が入力される論理ブロック 100 に入力され、結果としてレジスタ (W-CC) 108 の特定のビットを "1" にセットするセット信号 SET-CC を出力する。ちなみに、この場合のビットの指定は、カレントウインドウポインタ (CWP) 107 からの CWP0 ~ 4 で行なうことになる。

論理ブロック 100 からのセット信号 SET-CC とカレントウインドウポインタ (CWP) 107 からの CWP0 ~ 4 とによってビット指定されたレジスタ (W-CC) 108 は、ビット出力 ccn とビット出力 ccn-1 をカレントプロ

ックポインタ (CBP) 109 に送出する。ちなみに、ビット出力 ccn は現在のウインドウに対応するレジスタ (W-CC) 108 のビット状態であり、ビット出力 ccn-1 は前のウインドウに対応するレジスタ (W-CC) 108 のビット状態である。

カレントブロックポインタ (CBP) 109 は現在のウインドウが実際に使用する R-FILE 1 モジュール 101 でのブロックを指定する 4 ビットのバイナリカウンタであり、"-1"、"-2"、"+1"、"+2" の動作が可能である。このカレントブロックポインタ (CBP) 109 は、リセット信号 RESET が入力されると、プロシージャが最初に使用するウインドウに対して R-FILE 1 モジュール 101 のブロック 13 を割り当てるので、CWP = 13 となる。ちなみに、CWP = 0 の状態でディクリメント信号 DEC によりディクリメントすると CWP = 13 に戻る。このカレントブロックポインタ (CBP) 109 の出力 CWP-CBP0 ~ 3 に

よって、ウィンドウに割り当てるレジスタ数が制御される。なお、この出力CBP=CBP0~3はウィンドウのオーバーフロー/アンダーフローの情報を外部に知らせる機能をも有する。そして、CWPがCWP-1となって次のウィンドウに移るとき、ビット出力ccnが“1”ならCBPがCBP-2となり、現在のウィンドウに対して8本のレジスタ(2個のブロック)を割り当てる。一方、ビット出力ccn=“0”ならばCBPがCBP-1となり現在のウィンドウに対して4本のレジスタ(1個のブロック)を割り当てる。これに対して、CWPがCWP+1となり前のウィンドウに戻るときはビット出力ccn-1が“1”ならばCBPがCBP+2となり、ビット出力ccn-1が“0”ならばCBPがCBP+1となる。

サブストラクタ(INSAG)110, 111, 112は、レジスタアドレスがlasを指定したとき、las/outsのR-FILE1モジュール101へのブロック指定アドレスをoutsの値から

lasの値に変換する4ビットのサブストラクタであり、R-FILE1モジュール101のポートP0, P1, P2にそれぞれ対応して設けられる。そして、lasが指定されてビット出力ccn-1が“1”のときに、サブストラクタ(INSAG)110, 111, 112からの各出力信号CbP0'0~3, CbP1'0~3, CbP2'0~3の各値はCBP-2となり、ビット出力ccn-1が“0”の時にサブストラクタ(INSAG)110, 111, 112からの出力信号CbP0'0~3, CbP1'0~3, CbP2'0~3の値はCBP-1となる。一方、outsが指定されるとサブストラクタ(INSAG)110, 111, 112からの出力CbP0'0~3, CbP1'0~3, CbP2'0~3の各値としてはCBP-0~3がそのままスルーして出力される。

デコーダ(DEC)104, 105, 106はそれぞれレジスタアドレスrs1-A0~4, rs2-A0~4, rd-A0~4のうちの各上

位2ビットA3~4により、globals、las、localsのうちのいずれかのレジスタアドレスになっているかをデコードするものである。そして、A3=“0”、A4=“0”の時には、デコーダ(DEC)104からはglob-rs1が、デコーダ(DEC)105からはglob-rs2が、デコーダ(DEC)106からはglob-rdがそれぞれ出力される。また、A3=“1”、A4=“0”の時には、デコーダ(DEC)104からはln-rs1が、デコーダ(DEC)105からはln-rs2が、デコーダ(DEC)106からはln-rdがそれぞれ出力される。また、A3=“0”、A4=“1”の時には、デコーダ(DEC)104からはloc-rs1が、デコーダ(DEC)105からはloc-rs2が、デコーダ(DEC)106からはloc-rdがそれぞれ出力される。そして、A3=“1”、A4=“1”の時には、デコーダ(DEC)104からはout-rs1が、デコーダ(DEC)105からはout-rs2が、デコーダ(DEC)106からはout-rdがそれぞれ出力さ

れる。これらの信号のうちln-rs1、ln-rs2、ln-rdとout-rs1、rs2、rdとは論理ブロック200で処理され、LOSEL0~2というR-FILE1モジュール101の各ポートをイネーブルにする信号として送出されて、R-FILE1モジュール101の入力端子S0~2に入力される。また、loc-rs1、loc-rs2、loc-rdはR-FILE1モジュール102の各ポートをイネーブルにする信号として送出されR-FILE1モジュール102の入力端子S0~2に入力される。また、glob-rs1、glob-rs2、glob-rdはR-FILE1モジュール103の各ポートをイネーブルにする信号として送出されR-FILE1モジュール103の入力端子S0~2に入力される。ちなみに、ワーキングレジスタのアドレスはr0~7がglobalsに、r8~15がoutsに、r16~23がlocalsに、r24~31がlasにそれぞれ対応する。

以上のような構成において、次にその動作を第3図のタイミングチャートに基づいて説明する。

ちなみに、第3図(A)はクロックCLK、(B)はディクリメント信号DEC、(C)はインクリメント信号、(D)はR-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103に与えられるリードイネーブルREGR-EN、(E)はR-FILE1モジュール101、R-FILE1モジュール102およびR-FILE2モジュール103に与えられるライトイネーブルREGW-EN、(F)は各レジスタアドレスをレジスタ(W-CC)108にセットすることを許可する許可信号rs1CCW-EN、rs2CCW-EN、rbCCW-EN、(G)はレジスタアドレスrs1-A0~4、(H)はレジスタアドレスrs2-A0~4、(I)はレジスタアドレスrd-A0~4、(J)はソースレジスタ1の出力データrs1-D0~31、(K)はソースレジスタ2の出力データrs2-D0~31、(L)はデステネーションレジスタの出力データrd-D0~31、(M)はカレントウィンドウポイン

タ(CWP)107の出力CWP0~4、(N)はレジスタ(W-CC)108のビット出力ccn、(O)はレジスタ(W-CC)108のビット出力ccn-1、(P)はカレントブロックポイント(CBP)109の出力であるCbP0~3、(Q)はサブストラクタ(INSAG)110から送出されるCbP0'0~3、(R)はサブストラクタ(INSAG)111から送出されるCbP1'0~3、(S)はサブストラクタ(INSAG)112から送出されるCbP2'0~3、(T)はデコーダ(DEC)104から出力されるglob-rs1、(U)デコーダ(DEC)104から出力されるla-rs1、(V)はデコーダ(DEC)104から出力されるloc-rs1、(W)はデコーダ(DEC)104から出力されるout-rs1、(X)は論理ブロック100から出力されるセット信号SET-CCである。

第3図に示した命令のシーケンスは①でADD、②でSAVE、③でADD、④でSAVEとなっ

ている。①ADDではrdのr7にrs1のr31とrs2のr12を加算して移す。次に、②SAVEでは次のウィンドウに移り、CWPをCWP-1とする。そして、③ADDではrdのr6にrs1のr16とrs2のr12を加算して移す。そして、次の④SAVEでセーブする。ちなみに、この場合[r31]には7が、[r12]には3が、[r16]には2がそれぞれ予め入っているものとする。文献(前記刊行物)によるSPARCチップでは命令を4段のパイプラインで実行しているので、本実施例ではこれに準じて4段のパイプラインとする。次に、第4図に基づいてパイプライン構造の概略を説明する。ここでは、ADD命令を例にとって各ステージでの処理を説明する。

まず、ADD命令をメモリ200から取り出し、インストラクションレジスタ(IR)202に書き込む(ステップF)。次に、インストラクションレジスタ(IR)202の中の命令をインストラクションデコーダ(IDEC)204に入力してこれをデコードする。そして、次の命令に対す

るアドレスを、アドレスジェネレータ(AG)206およびプログラムカウンタ(PC)208により発生する。レジスタアドレスrs1、rs2(2つのソースレジスタのデータ)をレジスタファイル(RF)210から読み出し、Aレジスタ212、Bレジスタ214に書き込む(ステップD)。

Aレジスタ212のデータとBレジスタ214のデータを演算ユニットALU216で加算して、結果を結果レジスタ(R)218に書き込む(ステップE)。

結果レジスタ(R)218のデータをrdの示すデステネーションレジスタに書き込む(ステップS)。

第3図のタイミングチャートでは上記各ステップをクロックCLKのt1、t2、t3、t4に対応づけて示している。即ち、先ず、①のADDにおいては、t2のステップDでレジスタアドレスrs1-A0~4-31、レジスタアドレスrs2-A0~12となりレジスタファイルから



r31とr12のデータを読み出す。但し、実際の読み出し動作はt2の後半で行なっているため、データrs1-D0~31、データrs2-D0~31にデータが出力されるのはt2の後半となる。r31のデータは3であるので、データrs1-D0~31=7、データrs2-D0~31=3となる。そして、t3のステップEではr31+r12=10を実行し、結果をレジスタに書き込む。r31はinsレジスタなので、r31の読み出しのときにはCBP-1でCBP0'0~3-0となる。次の、t4のステップSでは、結果レジスタのデータをデステーションレジスタのr7に書き込む。書き込みはt4の前半で行なう。なお、t4の前半においては、③のADD命令のためにソースレジスタからの読み出しが行なわれている。

②のSAVEでは、t4のステップEでCWP=CWP-1を実行する。①の実行において、localsへのアクセスがなく且つoutsへのアクセスにおいてもr8~r15の中で後半のr12へのア

クセスであったので、レジスタ(W-CC)108のビット13="0"のままであるので、つまりウィンドウ13は4本モードのままであるので、CBP=CBP-1を実行する。

そして、③のADDでは、t4のステップDにおいて、r16とr12のデータ2,3がデータrs1-D0~31、rs2-D0~31に出力される。このとき、r16はlocalsの前半のレジスタなので、セット出力SET-CC="1"となり、レジスタ(W-CC)108のビット12に"1"がセットされる。その結果、ウィンドウ12は8本モードになる。

次の、④のSAVEでは、t4のステップEで、CWP=CWP-1を実行する。ウィンドウ12は8本モードになっているので、CBP=CBP-1を実行する。

以上の例では、localsとoutsの割り当て数をブロック単位で管理しており、localsとins/outsのレジスタ数が等しく、さらにlocalsとins/outsを分離して別のモジュールにしている。その結果、

ウィンドウ1個当たりのコンディション情報が1ビットでよく、実アドレス発生回路が簡単になり、実アドレスの計算も非常に高速に実行できるようになるというメリットがある。

なお、上記実施例では8本と4本の切り替えを例示したが、レジスタの割り当て数は8本~0本の範囲で任意に選ぶことができる。また、localsとoutsへのレジスタの割り当て数を異なるようにしても良い。

ちなみに、ウィンドウに対するレジスタ数を可変とすれば、レジスタ(W-CC)108としては14ビットレジスタが、カレントブロックポイント(CBP)109としては4ビットカウンタが、およびサブストラクタ(INSAG)110、111、112としては4ビットフルアダー×3=12ビットフルアダーの機能回路がそれぞれ必要となる。これに対し、ウィンドウをn個追加するのに必要な回路は、32ビット長レジスタ×16×nであるので、第1図の構成のほうがコストパフォーマンスが非常に優れていることがわ

かる。

また、上記実施例ではハードウェアで8本/4本の切り替えを行なっているが、プロセッサ自身が切り替え命令を持つようにして、その命令を用いてソフトウェア上で切り替えを行なうようにしても良いことはもちろんである。

この場合、第1図に示すW-CCレジスタのような機能をはたすレジスタの内容を、所定の値に書き変える命令を、プロセッサの命令セットに設ければ良い。

#### (発明の効果)

本発明によれば、複数のワーキングレジスタを有するシステムにおいて、ウィンドウ(モジュール)に割り当てるレジスタ数を変化することができるため、システムの冗長性が無くなり、ハードウェア資源を無駄無く有効に利用することが可能となり、コストパフォーマンスの優れた、レジスタウィンドウ方式によるマイクロプロセッサを得ることができるという可変方式の特長に加え、さらに、本発明の第1の発明によれば、レジスタ

ウィンドウのレジスタの割り当て数の記憶を、ワーキングレジスタに対するアドレス情報に基づいて自動的に行うようにしたので、固定方式の既存のプログラムをそのまま実行でき、かつウィンドウのオーバーフロー／アンダーフローによるオーバーヘッドを低減することの可能なマイクロプロセッサを得ることができる。

また、本発明の第2の発明によれば、前記割り当て数の記憶を、CPUが実行できる命令の1つに基づいて行うようにしたので、プログラム内のプロシージャの使用するレジスタの規模に応じて割り当て数の変更等をソフトウェアによって実現でき、従来技術による可変方式のプロセッサよりもさらにレジスタの使用効率を高めることができ、より使い勝手が良く、よりコストパフォーマンスの高い可変方式のウィンドウによるマイクロプロセッサを提供することができる。

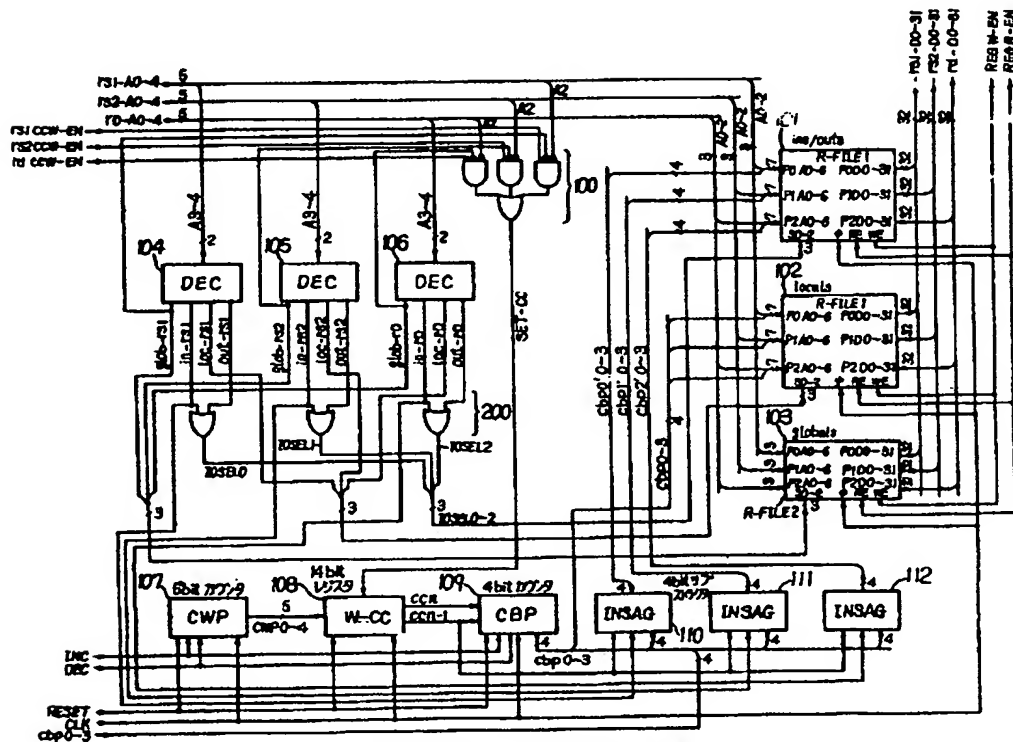
セッサのブロック図、第2図は第1図のR-FILE1モジュール101の詳細例を示す説明図、第3図は第1図の装置の動作を説明するタイミングチャート、第4図はパイプライン構造の概略説明図である。

- 101…R-FILE1モジュール、
- 102…R-FILE1モジュール、
- 103…R-FILE2モジュール、
- 104, 105, 106…デコーダ(DEC)、
- 107…カレントウィンドウポインタ(CWP)、
- 108…レジスタ(W-CC)、
- 109…カレントブロックポインタ(CBP)、
- 110, 111, 112…サブストラクタ(INSAG)。

#### 4. 図面の簡単な説明

第1図は本発明の一実施例に係るマイクロプロ

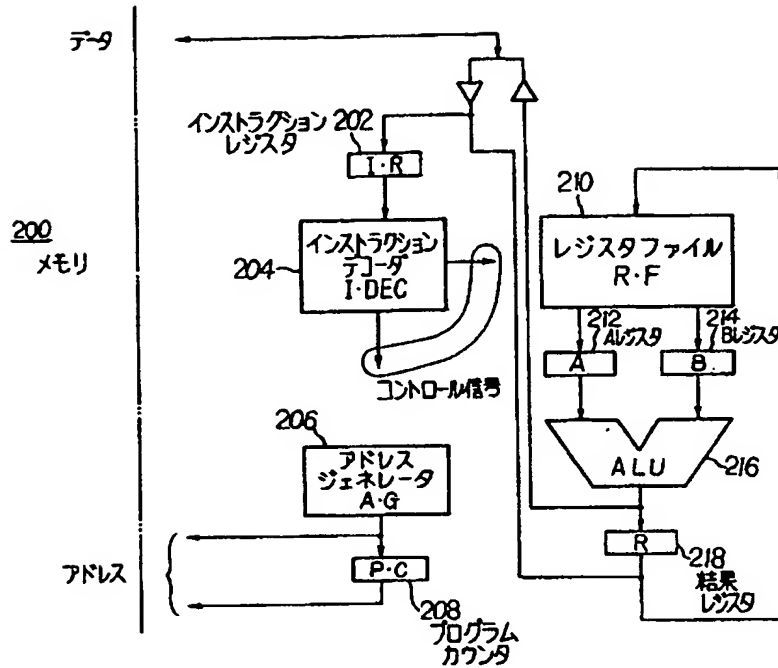
出願人代理人 佐 藤 一 雄



本発明の一実施例のブロック図

第1図





パイプライン構造の概略説明図  
第4図

手続補正書

平成 2 年 2 月 1 日

特許庁長官 吉田 文 教 閣

1 事件の表示

平成 1 年特許願第 37805 号

2 発明の名称

マイクロプロセッサ

3 補正をする者

事件との関係 特許出願人

(307) 株式会社 東 芝 (ほか 1 名)

4 代理人 (郵便番号 100)

東京都千代田区丸の内三丁目2番3号  
〔電話東京 (211) 2321 大代表〕

6428 弁理士 佐 藤 一

5 補正により する請求項の数

6 補正の対象

明細書の「特許請求の範囲」及び「発明の詳細な説明」の欄

8 補正の内容

- (1) 特許請求の範囲を別紙の通り訂正する。
- (2) 明細書第7頁下から3行目～第8頁下から5行目に記載の「本発明の第2…構成される。」を下記の通り補正する。

記

本願発明の第2のマイクロプロセッサは、複数個のレジスタウィンドウを有し、前記各レジスタウィンドウは複数個のレジスタを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウィンドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウィンドウのうちのいずれかを割り宛てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウィンドウへのレジスタ割り宛て数を、実行できる命令の1つに基づいて記憶す



る記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウィンドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えるものとして構成される。

特許請求の範囲

1. 複数個のレジスタウィンドウを有し、前記各レジスタウィンドウは複数個のレジスタを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウィンドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウィンドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウィンドウへのレジスタ割り当て数を、前記ワーキングレジスタに対するアドレス情報に基づいて自動的に記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウィンドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えることを特徴とするマイクロプロセッサ。

2. 複数個のレジスタウィンドウを有し、前記各レジスタウィンドウは複数個のレジスタを有するマイクロプロセッサであって、

前記各レジスタは前記レジスタウィンドウのうちの複数個のものに対して割り当てられており、さらに、

プログラム上の個々のプロシージャに対して前記レジスタウィンドウのうちのいずれかを割り当てることにより、前記各プロシージャ上で使用されるワーキングレジスタを決定する第1決定手段と、

前記各レジスタウィンドウへのレジスタ割り当て数を、実行できる命令の1つに基づいて記憶する記憶手段と、

前記記憶手段中の前記割り当て数に基づいて、前記各レジスタウィンドウが前記レジスタのうちのどれによって構成されるかを決定する第2決定手段と、を備えることを特徴とするマイクロプロセッサ。

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**